IN THE CLAIMS:

Please amend the claims as indicated below:

1. (Previously Presented) A method for programmatically generating a second graphical program based on a first graphical program, the method comprising:

receiving information specifying the first graphical program, wherein the first graphical program includes a first plurality of interconnected nodes which perform a first functionality, and wherein the first graphical program is associated with a first programming development environment;

constructing an abstract representation of the first graphical program based on the received information specifying the first graphical program; and

programmatically generating the second graphical program based on the constructed abstract representation, wherein the second graphical program implements functionality of the first graphical program, wherein the second graphical program includes a second plurality of interconnected nodes which perform the first functionality, wherein the second graphical program is associated with a second programming development environment, and wherein the second programming development environment is different from the first programming development environment.

2. (Original) The method of claim 1, wherein the second graphical program performs substantially like the first graphical program.

3. (Cancelled)

4. (Original) The method of claim 1,

wherein the second graphical program implements only a portion of functionality of the first graphical program.

5. (Cancelled)

6. (Previously Presented) The method of claim 1, wherein the first plurality of interconnected nodes in the first graphical program visually indicate the first functionality of the first graphical program;

wherein the second plurality of interconnected nodes in the second graphical program visually indicate the first functionality of the second graphical program.

7. (Previously Presented) The method of claim 1,

wherein the first plurality of interconnected nodes are interconnected according to data flow;

wherein the second plurality of interconnected nodes are interconnected according to data flow.

8. (Original) The method of claim 1,

wherein the first graphical program includes a first block diagram and a first user interface;

wherein the second graphical program includes a second block diagram and a second user interface.

9. (Original) The method of claim 8,

wherein the second block diagram appears substantially like the first block diagram.

10. (Original) The method of claim 8,

wherein the second user interface appears substantially like the first user interface.

11. (Cancelled)

12. (Cancelled)

13. (Original) The method of claim 1,

wherein the first graphical program is developed according to a first graphical programming language;

wherein the second graphical program is programmatically generated according to a second graphical programming language, wherein the second graphical programming language is different than the first graphical programming language.

14. (Original) The method of claim 13,

wherein the first graphical programming language is the G language.

15. (Original) The method of claim 13,

wherein the second graphical programming language is the G language.

16. (Original) The method of claim 1,

wherein the first graphical program is one of a data flow program, control flow program, or an execution flow program;

wherein the second graphical program is one of a data flow program, control flow program, or an execution flow program.

17. (Original) The method of claim 1, further comprising:

executing the second graphical program, wherein the second graphical program performs functionality of the first graphical program.

18. (Original) The method of claim 1, further comprising:

compiling the second graphical program;

downloading the second graphical program to a hardware device.

19. (Cancelled)

20. (Original) The method of claim 1,

wherein the first graphical program is one of a Simulink program, a VEE program, or a LabVIEW program.

21. (Original) The method of claim 1,

wherein the second graphical program is one of a Simulink program, a VEE program, or a LabVIEW program.

22. (Original) The method of claim 1,

wherein the received information specifying the first graphical program comprises one or more of text information and binary information.

23. (Original) The method of claim 1,

wherein the received information specifying the first graphical program comprises abstract information describing the first graphical program.

24. (Original) The method of claim 1,

wherein the received information specifying the first graphical program comprises a model file describing the first graphical program.

25. (Cancelled)

26. (Previously Presented) The method of claim 1,

wherein said constructing an abstract representation of the first graphical program comprises constructing a directed graph representation of the first graphical program.

27. (Previously Presented) The method of claim 1,

wherein said constructing an abstract representation of the first graphical program comprises constructing one or more data structures representing the first graphical program.

28. (Original) The method of claim 1,

wherein the first graphical program includes a first node;

wherein said receiving information specifying the first graphical program comprises receiving information specifying the first node;

wherein said programmatically generating the second graphical program comprises programmatically including a second node corresponding to the first node in the second graphical program.

29. (Original) The method of claim 28,

wherein the first node has particular functionality;

wherein the second node has functionality equivalent to the particular functionality of the first node.

30. (Previously Presented) A memory medium for programmatically generating a second graphical program based on a first graphical program, the memory medium comprising program instructions executable by a processor to:

receive information specifying the first graphical program, wherein the first graphical program includes a first plurality of interconnected nodes which perform a first functionality, and wherein the first graphical program is associated with a first programming development environment;

construct an abstract representation of the first graphical program based on the received information specifying the first graphical program; and

programmatically generate the second graphical program based on the constructed abstract representation, wherein the second graphical program implements functionality of the first graphical program, wherein the second graphical program includes a second plurality of interconnected nodes which perform the first functionality, wherein the second graphical program is associated with a second programming development environment, and wherein the second programming development environment is different from the first programming development environment.

31. (Original) The memory medium of claim 30, wherein the second graphical program performs substantially like the first graphical program.

32. (Cancelled)

33. (Original) The memory medium of claim 30,

wherein the second graphical program implements only a portion of functionality of the first graphical program.

34. (Cancelled)

35. (Previously Presented) The memory medium of claim 30,

wherein the first plurality of interconnected nodes are interconnected according to one or more of data flow, control flow, and execution flow; and

wherein the second plurality of interconnected nodes are interconnected according to one or more of data flow, control flow, and execution flow.

36. (Previously Presented) The memory medium of claim 30,

wherein the first graphical program includes a first block diagram and a first user interface; and

wherein the second graphical program includes a second block diagram and a second user interface.

37. (Cancelled)

38. (Previously Presented) A system for programmatically generating a second graphical program based on a first graphical program, the system comprising:

a processor coupled to a memory, wherein the memory stores program instructions;

wherein the processor is operable to execute the program instructions in order to:

receive information specifying the first graphical program, wherein the first graphical program includes a first plurality of interconnected nodes which perform a

first functionality, and wherein the first graphical program is associated with a first programming development environment;

construct an abstract representation of the first graphical program based on the received information specifying the first graphical program; and

programmatically generate the second graphical program based on the constructed abstract representation, wherein the second graphical program implements functionality of the first graphical program, wherein the second graphical program includes a second plurality of interconnected nodes which perform the first functionality, wherein the second graphical program is associated with a second programming development environment, and wherein the second programming development environment is different from the first programming development environment.

39 – 44 (Cancelled)

45. (Previously Presented) The method of claim 1, wherein the information specifying the first graphical program comprises the first graphical program.

46. (Previously Presented) The method of claim 1, wherein the first graphical program is a first type of data flow diagram;

wherein the second graphical program is a second type of data flow diagram.

47. (Previously Presented) The method of claim 1, wherein the first graphical program comprises one or more loops among block diagram node interconnections; and

wherein the second graphical program does not comprise one or more loops among block diagram node interconnections.

48 - 56 (Cancelled)

57. (Previously Presented) The method of claim 47, wherein the second graphical program comprises one or more structure nodes each corresponding to one or more of the loops among the block diagram node interconnections of the first graphical program,

wherein each of the respective structure nodes in the second graphical program indicates looping of one or more nodes associated with the respective structure node.

58. (Previously Presented) The method of claim 1, wherein the first graphical program does not comprise one or more loops among block diagram node interconnections; and

wherein the second graphical program comprises one or more loops among block diagram node interconnections.

59. (Previously Presented) The method of claim 58, wherein the first graphical program comprises one or more structure nodes each corresponding to one or more of the loops among the block diagram node interconnections of the second graphical program, wherein each of the respective structure nodes in the first graphical program indicates looping of one or more nodes associated with the respective structure node.

60. (Previously Presented) The method of claim 1, wherein the first graphical program is a data flow diagram, wherein the first graphical program comprises a structure node that performs at least one of conditional looping, iteration, conditional branching, and sequencing of nodes in the data flow diagram.

61. (Previously Presented) The method of claim 1, wherein the second graphical program is a data flow diagram, wherein the second graphical program comprises a structure node that performs at least one of conditional looping, iteration, conditional branching, and sequencing of nodes in the data flow diagram.

62 (Previously Presented) The method of claim 1, wherein the first graphical program has first data flow semantics, wherein the second graphical program has second data flow semantics, and wherein the second data flow semantics is different from the first data flow semantics.

63. (Previously Presented) The memory medium of claim 36, wherein the second block diagram appears substantially like the first block diagram.

64. (Previously Presented) The memory medium of claim 36, wherein the second user interface appears substantially like the first user interface.

65. (Previously Presented) The memory medium of claim 30, wherein the first graphical program is developed according to a first graphical programming language; and

wherein the second graphical program is programmatically generated according to a second graphical programming language, and wherein the second graphical programming language is different than the first graphical programming language.

66. (Previously Presented) The memory medium of claim 65, wherein the first graphical programming language is the G language.

67. (Previously Presented) The memory medium of claim 65, wherein the second graphical programming language is the G language.

68. (Previously Presented) The memory medium of claim 30, wherein the first graphical program is a data flow program; and

wherein the second graphical program is a data flow program.

69. (Previously Presented) The memory medium of claim 30, wherein the program instructions are further executable to:

execute the second graphical program, wherein the second graphical program performs functionality of the first graphical program.

70. (Previously Presented) The memory medium of claim 30, wherein the program instructions are further executable to:

compile the second graphical program; and

download the second graphical program to a hardware device.

71. (Previously Presented) The memory medium of claim 30, wherein the first graphical program is one of a Simulink program, a VEE program, or a LabVIEW program.

72. (Previously Presented) The memory medium of claim 30, wherein the second graphical program is one of a Simulink program, a VEE program, or a LabVIEW program.

73. (Previously Presented) The memory medium of claim 30, wherein the received information specifying the first graphical program comprises one or more of text information and binary information.

74. (Previously Presented) The memory medium of claim 30, wherein the received information specifying the first graphical program comprises a model file describing the first graphical program.

75. (Previously Presented) The memory medium of claim 30, wherein in said constructing an abstract representation of the first graphical program the program instructions are further executable to construct a directed graph representation of the first graphical program.

76. (Previously Presented) The memory medium of claim 30, wherein in said constructing an abstract representation of the first graphical program the program instructions are further executable to construct one or more data structures representing the first graphical program.

77. (Previously Presented) The memory medium of claim 30,

wherein the first graphical program includes a first node;

wherein in said receiving information specifying the first graphical program the program instructions are executable to receive information specifying the first node; and

wherein in said programmatically generating the second graphical program the program instructions are further executable to programmatically include a second node corresponding to the first node in the second graphical program.

78. (Previously Presented) The memory medium of claim 77,

wherein the first node has particular functionality; and

wherein the second node has functionality equivalent to the particular functionality of the first node.

79. (Previously Presented) The memory medium of claim 30, wherein the information specifying the first graphical program comprises the first graphical program.

80. (Previously Presented) The memory medium of claim 30, wherein the first graphical program is a first type of data flow diagram; and

wherein the second graphical program is a second type of data flow diagram.

81. (Previously Presented) The memory medium of claim 30, wherein the first graphical program comprises one or more loops among block diagram node interconnections; and

wherein the second graphical program does not comprise one or more loops among block diagram node interconnections.

82. (Previously Presented) The memory medium of claim 81, wherein the second graphical program comprises one or more structure nodes each corresponding to one or more of the loops among the block diagram node interconnections of the first graphical program, wherein each of the respective structure nodes in the second graphical program indicates looping of one or more nodes associated with the respective structure node.

83. (Previously Presented) The memory medium of claim 30, wherein the first graphical program does not comprise one or more loops among block diagram node interconnections; and

wherein the second graphical program comprises one or more loops among block diagram node interconnections.

84. (Previously Presented) The memory medium of claim 83, wherein the first graphical program comprises one or more structure nodes each corresponding to one or more of the loops among the block diagram node interconnections of the second graphical program, wherein each of the respective structure nodes in the first graphical program indicates looping of one or more nodes associated with the respective structure node.

85. (Previously Presented) The memory medium of claim 30, wherein the first graphical program is a data flow diagram, wherein the first graphical program comprises a structure node that performs at least one of conditional looping, iteration, conditional branching, and sequencing of nodes in the data flow diagram.

86. (Previously Presented) The memory medium of claim 30, wherein the second graphical program is a data flow diagram, wherein the second graphical program comprises a structure node that performs at least one of conditional looping, iteration, conditional branching, and sequencing of nodes in the data flow diagram.

87. (Previously Presented) A method for programmatically generating a second graphical program based on a first graphical program, the method comprising:

receiving information specifying the first graphical program, wherein the first graphical program includes a first plurality of interconnected nodes which perform a first functionality, and wherein the first graphical program is associated with a first programming development environment; and

programmatically generating the second graphical program based on the information, wherein the second graphical program implements functionality of the first graphical program, wherein the second graphical program includes a second plurality of interconnected nodes which perform the first functionality, wherein the second graphical program is associated with a second programming development environment, and

wherein the second programming development environment is different from the first programming development environment;

wherein the first graphical program comprises one or more loops among block diagram node interconnections; and

wherein the second graphical program does not comprise one or more loops among block diagram node interconnections.

88. (Previously Presented) The method of claim 87, wherein the second graphical program performs substantially like the first graphical program.

89. (Previously Presented) The method of claim 87,

wherein the second graphical program implements only a portion of functionality of the first graphical program.

90. (Previously Presented) The method of claim 87, wherein the first plurality of interconnected nodes in the first graphical program visually indicate the first functionality of the first graphical program; and

wherein the second plurality of interconnected nodes in the second graphical program visually indicate the first functionality of the second graphical program.

91. (Previously Presented) The method of claim 87,

wherein the first plurality of interconnected nodes are interconnected according to data flow semantics; and

wherein the second plurality of interconnected nodes are interconnected according to data flow semantics.

92. (Previously Presented) The method of claim 87,

wherein the first graphical program includes a first block diagram and a first user interface; and

wherein the second graphical program includes a second block diagram and a second user interface.

93. (Previously Presented) The method of claim 92,

wherein the second block diagram appears substantially like the first block diagram.

94. (Previously Presented) The method of claim 92,

wherein the second user interface appears substantially like the first user interface.

95. (Previously Presented) The method of claim 87,

wherein the first graphical program is developed according to a first graphical programming language; and

wherein the second graphical program is programmatically generated according to a second graphical programming language, and wherein the second graphical programming language is different than the first graphical programming language.

96. (Previously Presented) The method of claim 95,

wherein the second graphical programming language is the G language.

97. (Previously Presented) The method of claim 87,

wherein the first graphical program is one of a data flow program, control flow program, or an execution flow program; and

wherein the second graphical program is one of a data flow program, control flow program, or an execution flow program.

98. (Previously Presented) The method of claim 87, further comprising:

executing the second graphical program, wherein the second graphical program performs functionality of the first graphical program.

99. (Previously Presented) The method of claim 87, further comprising:

compiling the second graphical program; and

downloading the second graphical program to a hardware device.

100. (Previously Presented) The method of claim 87,

wherein the first graphical program is one of a Simulink program or a VEE program.

101. (Previously Presented) The method of claim 87,

wherein the second graphical program is one of a Simulink program, a VEE program, or a LabVIEW program.

102. (Previously Presented) The method of claim 87,

wherein the received information specifying the first graphical program comprises one or more of text information and binary information.

103. (Previously Presented) The method of claim 87,

wherein the received information specifying the first graphical program comprises abstract information describing the first graphical program.

104. (Previously Presented) The method of claim 87,

wherein the received information specifying the first graphical program comprises a model file describing the first graphical program.

105. (Previously Presented) The method of claim 87, further comprising:

constructing an abstract representation of the first graphical program based on the received information specifying the first graphical program;

wherein said programmatically generating the second graphical program comprises programmatically creating the second graphical program based on the constructed abstract representation.

106. (Previously Presented) The method of claim 105,

wherein said constructing an abstract representation of the first graphical program comprises constructing a directed graph representation of the first graphical program.

107. (Previously Presented) The method of claim 105,

wherein said constructing an abstract representation of the first graphical program comprises constructing one or more data structures representing the first graphical program.

108. (Previously Presented) The method of claim 87,

wherein the first graphical program includes a first node;

wherein said receiving information specifying the first graphical program comprises receiving information specifying the first node; and

wherein said programmatically generating the second graphical program comprises programmatically including a second node corresponding to the first node in the second graphical program.

109. (Previously Presented) The method of claim 108,

wherein the first node has particular functionality; and

wherein the second node has functionality equivalent to the particular functionality of the first node.

110. (Previously Presented) The method of claim 87, wherein the information specifying the first graphical program comprises the first graphical program.

111. (Previously Presented) The method of claim 87, wherein the first graphical program is a first type of data flow diagram; and

wherein the second graphical program is a second type of data flow diagram.

112. (Previously Presented) The method of claim 87, wherein the second graphical program comprises one or more structure nodes each corresponding to one or more of the loops among the block diagram node interconnections of the first graphical program, wherein each of the respective structure nodes in the second graphical program indicates looping of one or more nodes associated with the respective structure node.

113. (Previously Presented) The method of claim 87, wherein the second graphical program is a data flow diagram, wherein the second graphical program comprises a structure node that performs at least one of conditional looping, iteration, conditional branching, and sequencing of nodes in the data flow diagram.

114. (Previously Presented) The method of claim 87, wherein the first graphical program has first data flow semantics, wherein the second graphical program has second data flow semantics, and wherein the second data flow semantics is different from the first data flow semantics.

115. (Previously Presented) A memory medium for programmatically generating a second graphical program based on a first graphical program, the memory medium comprising program instructions executable by a processor to:

receive information specifying the first graphical program, wherein the first graphical program includes a first plurality of interconnected nodes which perform a first functionality, and wherein the first graphical program is associated with a first programming development environment; and

programmatically generate the second graphical program based on the information, wherein the second graphical program implements functionality of the first graphical program, wherein the second graphical program includes a second plurality of interconnected nodes which perform the first functionality, wherein the second graphical program is associated with a second programming development environment, and wherein the second programming development environment is different from the first programming development environment;

wherein the first graphical program comprises one or more loops among block diagram node interconnections; and

wherein the second graphical program does not comprise one or more loops among block diagram node interconnections.

116. (Previously Presented) The memory medium of claim 115, wherein the second graphical program performs substantially like the first graphical program.

117. (Previously Presented) The memory medium of claim 115,

wherein the second graphical program implements only a portion of functionality of the first graphical program.

118. (Previously Presented) The memory medium of claim 115, wherein the first plurality of interconnected nodes in the first graphical program visually indicate the first functionality of the first graphical program; and

wherein the second plurality of interconnected nodes in the second graphical program visually indicate the first functionality of the second graphical program.

119. (Previously Presented) The memory medium of claim 115,

wherein the first plurality of interconnected nodes are interconnected according to one or more of data flow, control flow, and execution flow; and

wherein the second plurality of interconnected nodes are interconnected according to one or more of data flow, control flow, and execution flow.

120. (Previously Presented) The memory medium of claim 115,

wherein the first graphical program includes a first block diagram and a first user interface; and

wherein the second graphical program includes a second block diagram and a second user interface.

121. (Previously Presented) The memory medium of claim 120,

wherein the second block diagram appears substantially like the first block diagram.

122. (Previously Presented) The memory medium of claim 120,

wherein the second user interface appears substantially like the first user interface.

123. (Previously Presented) The memory medium of claim 115,

wherein the first graphical program is developed according to a first graphical programming language; and

wherein the second graphical program is programmatically generated according to a second graphical programming language, wherein the second graphical programming language is different than the first graphical programming language.

124. (Previously Presented) The memory medium of claim 123,

wherein the second graphical programming language is the G language.

125. (Previously Presented) The memory medium of claim 115,

wherein the first graphical program is one of a data flow program, control flow program, or an execution flow program; and

wherein the second graphical program is one of a data flow program, control flow program, or an execution flow program.

126. (Previously Presented) The memory medium of claim 115, wherein the program instructions are further executable to:

execute the second graphical program, wherein the second graphical program performs functionality of the first graphical program.

127. (Previously Presented) The memory medium of claim 115, wherein the program instructions are further executable to:

compile the second graphical program; and

download the second graphical program to a hardware device.

128. (Previously Presented) The memory medium of claim 115,

wherein the first graphical program is one of a Simulink program or a VEE program.

129. (Previously Presented) The memory medium of claim 115,

wherein the second graphical program is one of a Simulink program, a VEE program, or a LabVIEW program.

130. (Previously Presented) The memory medium of claim 115,

wherein the received information specifying the first graphical program comprises one or more of text information and binary information.

131. (Previously Presented) The memory medium of claim 115,

wherein the received information specifying the first graphical program comprises abstract information describing the first graphical program.

132. (Previously Presented) The memory medium of claim 115,

wherein the received information specifying the first graphical program comprises a model file describing the first graphical program.

133. (Previously Presented) The memory medium of claim 115, wherein the program instructions are further executable to:

construct an abstract representation of the first graphical program based on the received information specifying the first graphical program;

wherein in said programmatically generating the second graphical program the program instructions are further executable to programmatically create the second graphical program based on the constructed abstract representation.

134. (Previously Presented) The memory medium of claim 133,

wherein in said constructing an abstract representation of the first graphical program the program instructions are further executable to construct a directed graph representation of the first graphical program.

135. (Previously Presented) The memory medium of claim 133,

wherein in said constructing an abstract representation of the first graphical program the program instructions are further executable to construct one or more data structures representing the first graphical program.

136. (Previously Presented) The memory medium of claim 115,

wherein the first graphical program includes a first node;

wherein in said receiving information specifying the first graphical program the program instructions are further executable to receive information specifying the first node; and

wherein in said programmatically generating the second graphical program the program instructions are further executable to programmatically include a second node corresponding to the first node in the second graphical program.

137. (Previously Presented) The memory medium of claim 136,

wherein the first node has particular functionality; and

wherein the second node has functionality equivalent to the particular functionality of the first node.

138. (Previously Presented) The memory medium of claim 115, wherein the information specifying the first graphical program comprises the first graphical program.

139. (Previously Presented) The memory medium of claim 115, wherein the first graphical program is a first type of data flow diagram; and

wherein the second graphical program is a second type of data flow diagram.

140. (Previously Presented) The memory medium of claim 115, wherein the second graphical program comprises one or more structure nodes each indicating control flow corresponding to one or more of the loops among the block diagram node interconnections of the first graphical program.

141. (Previously Presented) The memory medium of claim 115, wherein the second graphical program is a data flow diagram, wherein the second graphical program comprises a structure node that performs at least one of conditional looping, iteration, conditional branching, and sequencing of nodes in the data flow diagram.

142. (Previously Presented) A system for programmatically generating a second graphical program based on a first graphical program, the system comprising:

a processor coupled to a memory, wherein the memory stores program instructions;

wherein the processor is operable to execute the program instructions in order to:

receive information specifying the first graphical program, wherein the first graphical program includes a first plurality of interconnected nodes which perform a first functionality, and wherein the first graphical program is associated with a first programming development environment; and

programmatically generate the second graphical program based on the information, wherein the second graphical program implements functionality of the first graphical program, wherein the second graphical program includes a second plurality of interconnected nodes which perform the first functionality, wherein the second graphical program is associated with a second programming development environment, and wherein the second programming development environment is different from the first programming development environment;

wherein the first graphical program comprises one or more loops among block diagram node interconnections; and

wherein the second graphical program does not comprise one or more loops among block diagram node interconnections.

143.-198. (Cancelled)